INSTRUCTION MANUAL

Model 747
Device Controller

Serial Number: ____________________

– A Schoeffel Group Company –

McPHERSON™

7A Stuart Road • Chelmsford, Massachusetts 01824
PHONE: 978-256-4512 / 1-800-255-1055
FAX: 978-250-8625
EMAIL: MCP@mcphersoninc.com
WWW: http://www.mcphersoninc.com

TABLE OF CONTENTS

Model 747 Device Controller

1.0 Introduction

This manual describes the operation, control and communication parameters for the 747 Series Device Controller. It also helps you to understand the system operation characteristics.

Since we constantly try to improve our product line, we occasionally issue addenda or appendix that document additional features, changes and warranties which affect our controllers. If addenda or an appendix is included with this manual, please read it to see if it affects your particular controller or warranty.

1.1 Controller Description

The McPherson Model 747 Series Device Controller is a PLC (Programmable Logic Controller) based motion controller which has been custom fitted with McPherson Software to work with various precision accessories used in automation of spectrometers.

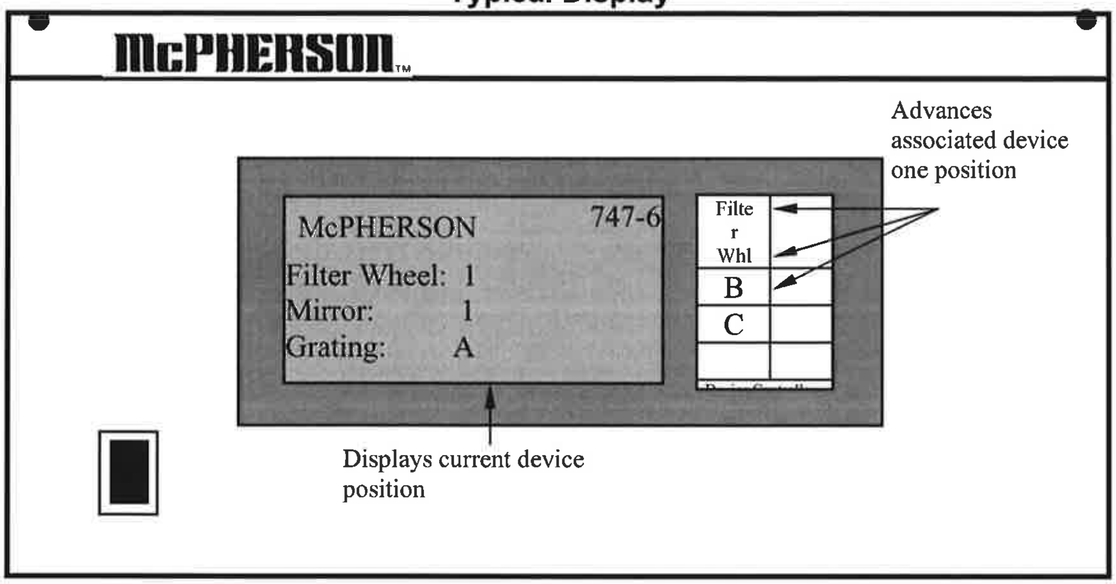
The 747 Series Controller is comprised of a CPU and associated software which supports eight (8) Input points and eight (8) Output points. The controller has been programmed to allow for local operation via the front panel of the controller, however, limited remote operation is possible and will be discussed later on.

1.2 Operating Parameters

Input Voltage Range	85 – 264 VAC
	47 – 63 Hz
Power Consumption	50 VA
Inrush Current Max.	30 A
Temperature	32° to 131° F (-20° to 70° C)

1.3 Controls and Display

Typical Display



1.4 Local Operation

1. Connect the device assembly to the "auxiliary" connection of the controller.

2. Connect the power cord to the controller's power source.

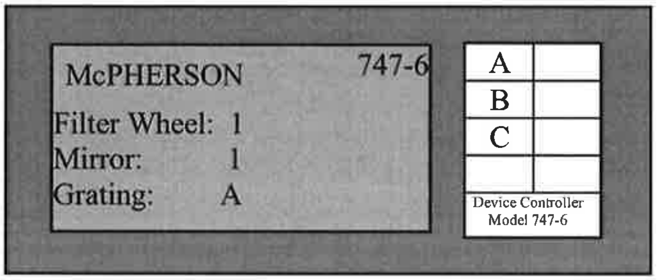
3. Set the power switch to the "ON" position. After a few seconds, the controller will display the screen shown in Section 1.3.

4. Select the device and press the associated button. The device will first attempt to find its "Home" position (either position "1" or "A" depending on the device selected) then proceed to 1 position beyond its last known position.

Note: In the event a home position is not found, i.e., device is not connected, a Timeout Error will be displayed. The error will remain displayed until the problem has been corrected.

5. Continue to press the button until the desired position is reached.

6. After initialization has been completed, the device will be in position #1 as displayed below:



5. To move to the next position, press and release the move button. The filter wheel assembly will now go to the next position and update the display appropriately. The controller increments only in single steps. Therefore, continue to press the move button (after motion has stopped) until the desired position is reached.

1.5 Remote Operation

Remote control of the 747 Device Controller is a master/slave type of architecture. The master or host station must initiate requests to transfer data. McPherson has developed a wide range of software which can meet most of your needs in this area. However, if you are planning on developing your own program and if you are using a host computer as a master or host station, you are responsible for developing a communications program which follows the protocol detailed below.

Serial Communications Parameters

To establish communications with the 747 use a RS232 serial port with these settings: Baud Rate = 9600, Data Bits = 8, Stop Bits = 1, Parity = None, and Flow Control = None.

Communication Protocol

All communications with the 747 must follow the DirectNET protocol. Information given here covers control of a McPherson Model 747 Device Controller only. It is not intended as a reference for all communications functions of the PLC controller. All data is exchanged in ASCII format with the 747. For additional information refer to the Koyo DirectNET User Manual.

Messages consist of these protocol components:

Enquiry - initiates a request (from the master or host computer) with the slave stations.

Header -defines the operation as a read or write, the slave station address, and the type and amount of data to be transferred.

Data - the actual data that is being transferred.

Acknowledge - verifies that the communication is working properly.

End Of Transmission - indicates the communication is finished.

All read and write requests use ASCII control codes and a Longitudinal Redundancy Check (LRC) to manage the communications between the computer and the 747. The following control codes identify the beginning and end of protocol components such as enquiry, acknowledge, etc. The LRC is a checksum that ensures the data was transmitted and received correctly.

The symbol, Hex ASCII code, and a description of all control codes in communications between the Host and 747 are listed in Table 1.

Symbol	Hex ASCII Code	Description
<ENQ>	05	Enquiry - initiate request
<ACK>	06	Acknowledge - the communication was received without error
<NAK>	15	Negative Acknowledge - there was a problem with the communication
<SOH>	01	Start of Header - beginning of header
<ETB>	17	End of Transmission Block - end of immediate block
<STX>	02	Start of Text - beginning of data block
<ETX>	03	End of Text - End of last data block
<EOT>	04	End of Transmission

Table 1: ASCII control codes used to manage communications with the 747.

The diagram in Figure 1 shows the proper sequence for transmission of protocol components for both Read and Write operations. For remote control of the 747 the Host initiates all communications and must also terminate all transmissions. The individual protocol components are described in detail in the following sections.

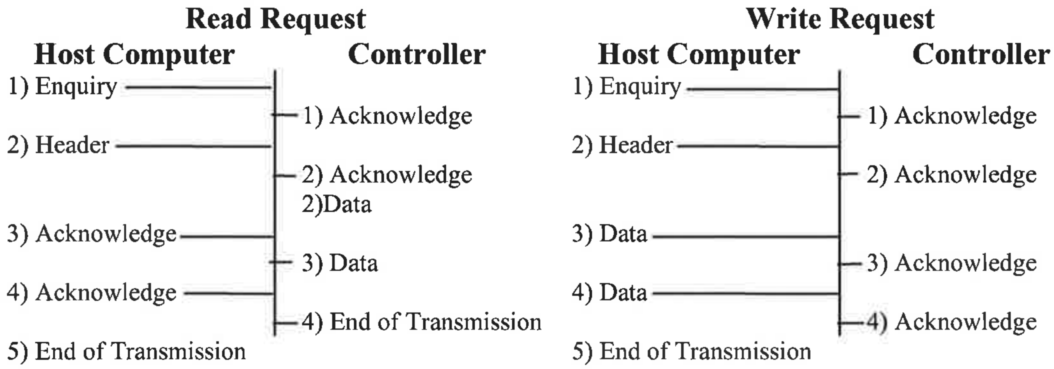


Figure 1: Message transmission sequence.

Enquiry

An Enquiry is a three-byte message that initiates the request with the controller. The message always begins with ASCII 4E ("N"), which means normal enquiry sequence. The second character contains the offset address of the 747; Hex 01 (default DirectNET address of the 747) plus an offset of HEX 20 which results in an offset address of Hex 21. The last character is the ASCII control code for Enquiry (05) which ends the request transmission block.

A complete, three-byte, Enquiry to a 747 at the default address is then: in Hex 4E2105 or in text N!<**ENQ**>.

Acknowledging the Request

The 747 will respond to an Enquiry with a three-character acknowledgement. The first two bytes of the Acknowledgment are the same as those of the Enquiry. They will be returned along with an <**ACK**> (ACKnowledge) if the information was transmitted and received without any problem. If problems were encountered then a <**NAK**> (Not AcKnowledge) will be returned as the third byte. If the 747 has not responded within 800 msec, then a timeout error should be generated by the Host.

For the complete Enquiry given above a successful transmission will be answered by: in Hex 4E2106 or in ASCII text N!<**ACK**>. If problems arise the response will be: in Hex 4E2115 or in text N!<**NAK**>.

If the Host does not send the Header within 800 msec, then a timeout will occur in the 747 and it will send an <**EOT**> to terminate communications. The Host must send an <**EOT**> to establish a new communication session with the 747.

Header

The Header is a 18-byte message that defines the requested operation. It is sent from the Host and contains the following information:

- Type of operation (read or write)
- Type of data being transferred
- Data address
- Number of complete data blocks
- Number of bytes in the last data block

For remote control of the 747 only V-Memory type data is accessed and only a few bytes at time. Table 2 defines a Header for the cases of writing four ASCII bytes to octal address 40600 and reading four ASCII bytes from octal address 2240.

Once the 747 has received the Header, it will Acknowledge receipt with either an <**ACK**> or <**NAK**> within 2000 msec or a timeout error should be generated by the Host.

	Byte #	Hex ASCII Code for Write	Hex ASCII Code for Read
Start of Header <**SOH**>	1	01	01
Controller Address (default)	2 & 3	3031	3031
Operation Read or Write	4	38	30
Data Type (V Memory)	5	31	31
Starting Memory Address (MSB)	6 & 7	3431	3034
Starting Memory Address (LSB)	8 & 9	3831	4131
Complete Data Blocks (None)	10 & 11	3030	3030
Partial Data Block (Four Bytes)	12 & 13	3034	3034
Host Computer Address	14 & 15	3031	3031
End of Transmission Block <**ETB**>	16	17	17
Checksum (LRC)	17 & 18	3031	3731

Table 2: Components of the Header transmission block.

Data

This section of the message contains the data being transferred between the host computer and the controller. In a Read operation data is sent from the 747 to the Host. In a Write operation data is sent from the Host to the 747. The data is transferred in full blocks of 256 bytes plus a final, partial block of less than 256 bytes. The 256-byte limit does not include control characters that signal the beginning or end of the data.

For remote control of the 747, only a single partial block of data is required. For transmission of this small amount of data, 800 msec is more than sufficient time and the Host should generate a timeout error if the data is not received within this period. The following table lists a transmission block which sends four ASCII bytes of data. The data is decimal one (1).

In the case of a read operation, the Host must validate the checksum (LRC) of the data sent from the 747 and Acknowledge receipt with an <**ACK**> or request re-transmission with a <**NAK**> within 800 msec.

Only V-Memory type data is exchanged between the Host and 747. V Memory is always accessed as two-byte words. The words are transmitted in ASCII format. A word is transmitted as four ASCII bytes. Each ASCII byte represents four bits of the word. The least significant byte of the word is transferred first as two ASCII bytes and the most significant byte of the word is transferred last as two ASCII bytes. Refer to Table 4.

	Byte #	Hex ASCII Code
Start of Text <**STX**>	1	02
Data – ASCII Byte 3	2	30
Data – ASCII Byte 4	3	31
Data – ASCII Byte 1	4	30
Data – ASCII Byte 2	5	30
End of Text <**ETX**>	6	03
Checksum (LRC)	7 & 8	3031

Table 3: Components of the Data section of a message.

V-Memory Word															
Most Significant Byte								Least Significant Byte							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ASCII Byte 1				ASCII Byte 2				ASCII Byte 3				ASCII Byte 4			
Order of transmission: ASCII Byte 3, ASCII Byte 4 ASCII Byte 1 ASCII Byte 2															

Table 4: Details of V-Memory Word transmission.

End of Transmission

The host computer must **always** end the communication by sending the End Of Transmission <**EOT**> control code; Hex ASCII 04. Even if the 747 generates a timeout error during transmission of a message and sends an <**EOT**> to the Host, it must be answered with an <**EOT**> from the Host.

For a Write operation, once the last data block has been transferred and acknowledged with an <**ACK**> from the 747 the transmission is terminated by the Host.

At the end of a Read operation, once the Host had acknowledged receipt of all data with an <**ACK**> the 747 will send an <**EOT**> within 800 msec or the Host should generate a timeout error. Once the Host has received an <**EOT**> from the 747, it must respond to the 747 with an <**EOT**> to prepare the 747 for the next message.

Longitudinal Redundancy Check (LRC)

For the relatively large blocks of data comprising the Header and Data portions of a message, a checksum called Longitudinal Redundancy Check (LRC) must be generated and included as the final two bytes in each block.

For a Header, the LRC is the exclusive OR of all bytes between the **SOH** and **ETB** control codes, i.e. bytes 2 - 16. For Data blocks the LRC is the exclusive OR of all bytes between the **STX** and **ETX** control codes.

V-Memory Locations

Just a handful of memory locations are accessed for remote control of the 747 and the four devices it controls. V Memory is always accessed as two-byte words with the address specified in octal. The needed addresses and bit locations of the various controls are detailed below.

The 747 Device Controller will only move one device at a time. It is possible to simultaneously set the data for moves of multiple devices by remote control, but is strongly discouraged.

The Hex ASCII reference addressed needed for the message header are found by converting the octal address to Hex and adding one. This value is then converted to Hex ASCII. For example, octal address 2240 is Hex 04A0. The Hex reference address is then 04A0 + 1 = 04A1. Converting to Hex ASCII results in 30 34 41 31. This example corresponds to the data listed in the Read column of Table 2.

Initialization Flags:

Octal Address: 40602 bits 0 - 3. Bits 0 – 3 correspond to devices 1 – 4, respectively. A "1" means the device must be initialized . Set the appropriate Increment Position Bit to initialize the device. A "0" indicates the device is ready. Devices require initialization after power up and following some error conditions.

In Motion Flags:

Octal Address: 40601 bits 16 – 20. Bit 16 reflects the motion status of all four devices. If any device is in motion this bit will contain a "1". Bits 17 – 20 correspond to devices 1 – 4, respectively. A "1" means the device is in motion . A "0" indicates the device is stopped. Poll these flags for completion of motion.

Error Flag:

Octal Address: 40600 bit 8. In case of a system error in the 747, bit 8 will be a "1". Poll this location to check for errors.

Increment Position Bits:

Octal Address: 40600 bits 0 – 3. Bits 0 – 3 correspond to devices 1 – 4, respectively. To initialize a device or increment the position by one, e.g. position 1 to position 2, set the appropriate bit to "1".

Current Positions:

Octal Addresses: 2240 – 2243. Addresses 2240 – 2243 correspond to devices 1 – 4, respectively. Read the value at the appropriate address to determine the current location of a device. This data is not valid if the device has not been initialized.

Destinations:

Octal Addresses: 2250 – 2253. Addresses 2240 – 2243 correspond to devices 1 – 4, respectively. To send a device to a desired position without using the Increment Position Bit multiple time, write the destination into the appropriate address. Caution: Do not write a value greater than the number of physical positions of the device. An error will be generated which will require the device to be initialized or the 747 power cycled.